**CSS Grid**

Create Your First CSS Grid

Turn any HTML element into a grid container by setting its display property to grid. This gives you the ability to use all the other properties associated with CSS Grid.

Note: In CSS Grid, the parent element is referred to as the container and its children are called items.

Change the display of the div with the container class to grid.

Add Columns with grid-template-columns

Simply creating a grid element doesn't get you very far. You need to define the structure of the grid as well.

To add some columns to the grid, use the grid-template-columns property on a grid container as demonstrated below:

.container {

display: grid;

grid-template-columns: 50px 50px;

}

This will give your grid two columns that are each 50px wide.

The number of parameters given to the grid-template-columns property indicates the number of columns in the grid, and the value of each parameter indicates the width of each column.

Add Rows with grid-template-rows

The grid you created in the last challenge will set the number of rows automatically. To adjust the rows manually, use the grid-template-rows property in the same way you used grid-template-columns in the previous challenge.

Add two rows to the grid that are 50px tall each.

grid-template-rows: 50px 50px;

Use CSS Grid units to Change the Size of Columns and Rows

You can use absolute and relative units like px and em in CSS Grid to define the size of rows and columns. You can use these as well:

* fr: sets the column or row to a fraction of the available space,
* auto: sets the column or row to the width or height of its content automatically,
* %: adjusts the column or row to the percent width of its container.

Here's the code that generates the output in the preview:

grid-template-columns: auto 50px 10% 2fr 1fr;

This snippet creates five columns. The first column is as wide as its content, the second column is 50px, the third column is 10% of its container, and for the last two columns; the remaining space is divided into three sections, two are allocated for the fourth column, and one for the fifth.

Create a Column Gap Using grid-column-gap

So far in the grids you have created, the columns have all been tight up against each other. Sometimes you want a gap in between the columns.

To add a gap between the columns, use the grid-column-gap property like this:

grid-column-gap: 10px;

This creates 10px of empty space between all of our columns.

Create a Row Gap using grid-row-gap

You can add a gap in between the rows of a grid using grid-row-gap in the same way that you added a gap in between columns in the previous challenge.

Create a gap for the rows that is 5px tall.

grid-row-gap: 5px;

Add Gaps Faster with grid-gap

grid-gap is a shorthand property for grid-row-gap and grid-column-gap from the previous two challenges that's more convenient to use. If grid-gap has one value, it will create a gap between all rows and columns. However, if there are two values, it will use the first one to set the gap between the rows and the second value for the columns.

Use grid-gap to introduce a 10px gap between the rows and 20px gap between the columns.

grid-gap: 10px 20px;

Use grid-column to Control Spacing

Up to this point, all the properties that have been discussed are for grid containers. The grid-column property is the first one for use on the grid items themselves.

The hypothetical horizontal and vertical lines that create the grid are referred to as lines.

These lines are numbered starting with 1 at the top left corner of the grid and move right for columns and down for rows, counting upward.

To control the number of columns an item will consume, you can use the grid-column property in conjunction with the line numbers you want the item to start and stop at.

Here's an example:

grid-column: 1 / 3;

This will make the item start at the first vertical line of the grid on the left and span to the 3rd line of the grid, consuming two columns.

Make the item with the class item5 consume the last two columns of the grid.

 grid-column:  2 / 4;

Use grid-row to Control Spacing

Of course, you can make items consume multiple rows just like you can with columns.

You define the horizontal lines you want an item to start and stop at using the grid-row property on a grid item.

Make the element with the item5 class consume the last two rows.

grid-row: 2 / 4;

Align an Item Horizontally using justify-self

In CSS Grid, the content of each item is located in a box which is referred to as a cell.

You can align the content's position within its cell horizontally using the justify-self property on a grid item.

By default, this property has a value of stretch, which will make the content fill the whole width of the cell. This CSS Grid property accepts other values as well:

* start: aligns the content at the left of the cell,
* center: aligns the content in the center of the cell,
* end: aligns the content at the right of the cell.

Use the justify-self property to center the item with the class item2.

justify-self: center;

Align an Item Vertically using align-self

Just as you can align an item horizontally, there's a way to align an item vertically as well.

To do this, you use the align-self property on an item. This property accepts all of the same values as justify-self from the last challenge.

Align the item with the class item3 vertically at the end.

align-self: end;

Align All Items Horizontally using justify-items

Sometimes you want all the items in your CSS Grid to share the same alignment.

You can use the previously learned properties and align them individually, or you can align them all at once horizontally by using justify-items on your grid container.

This property can accept all the same values you learned about in the previous two challenges, the difference being that it will move all the items in our grid to the desired alignment.

Use this property to center all our items horizontally.

  justify-items: center;

Align All Items Vertically using align-items

Using the align-items property on a grid container will set the vertical alignment for all the items in our grid.

Use it now to move all the items to the end of each cell.

 align-items: end;

Divide the Grid Into an Area Template

You can group cells of your grid together into an area and give the area a custom name. Do this by using grid-template-areas on the container like this:

grid-template-areas:

"header header header"

"advert content content"

"advert footer footer";

The code above groups the cells of the grid into four areas;

* header, advert, content, footer.

Every word represents a cell and every pair of quotation marks represent a row.

Change the template so the footer area spans the entire bottom row.

Defining the areas won't have any visual effect right now. Later, you will make an item use an area to see how it works.

grid-template-areas:

      "header header header"

      "advert content content"

      "footer footer footer";